**CHAPTER 1**

**INTRODUCTION**

**1.1 GENERAL BACKGROUND**

Fake News Detection System offers detection of malicious image and news detection, image manipulation, since the domain of dictators and spy agencies has now become accessible to legions of common Internet trolls and Facebook conmen. With only rudimentary editing skills, it is now made possible to create realistic image composites, ﬁll in large image regions generate plausible video from speech etc. The system also provides several exploratory analyses on the identification of linguistic differences in fake and legitimate news content. One problem is that standard supervised learning approaches which have been very successful for many types of detection problems, are not well-suited for image forensics because the space of manipulated images is so vast and diverse, that it is rather unlikely we will ever have enough manipulated training data for a supervised method to fully succeed. In ‘Fake News Detection’, we turn to a vast and previously underutilized source of data, image EXIF metadata. EXIF tags are camera speciﬁcations that are digitally engraved into an image ﬁle at the moment of capture and are ubiquitously available.

**CHAPTER 2**

**LITERATURE SURVEY**

**STUDY OF SIMILAR WORKS**

**2.1 EXISTING SYSTEM**

In existing system, the prominence of disinformation is growing rapidly. Fake news is being created intentionally to misguide the readers. In existing system, we use Natural Language Processor to detect the fake news. NLP provides less accuracy of the news. The fake news and manipulated malicious images are spread through traditional news media and social media. The reliability of existing system is deteriorating on each passing day. The wide spread of fake news in existing system is having a huge negative impact on individuals and society.

Limitations:

* Wide spread of unreliable information.
* Inefficient and inflexible counter measures.
* Creating a negative outlook in the society.
* Lack of clean data will overshadow the original data while losing the value of data.

**CHAPTER 3**

**OVERALL DESCRIPTION**

**3.1 PROPOSED SYSTEM**

‘Fake News Detection’ system provides a platform by developing a machine learning program also by using Conventional Neural Network (CNN) to identify fake/unreliable news based on content acquired. CNN is a type of artificial neural network that uses perceptron. The proposed system also provides several exploratory analyses on the identification of linguistic differences in fake and legitimate news content. It also conducts a set of learning experiments to build accurate fake news detectors. With machine learning it gives a deep impression on the depth of fake news by deep learning processing, we propose to use the EXIF metadata as a supervisory signal for training a classiﬁcation model to determine whether an image is self-consistent – that is, whether different parts of the same image could have been produced by a single imaging pipeline. The model is self-supervised in that only real photographs and their EXIF meta-data are used for training. A consistency classiﬁer is learned for each EXIF tag separately using pairs of photographs, and the resulting classiﬁers are combined together to estimate self-consistency of pair so it patches in a novel input image.

The different modules included in fake news detection are as follows:

Dataset Collection: A data set (or dataset) is a collection of data. The data set lists values for each of the variables, such as height and weight of an object, for each member of the data set. Each value is known as a datum. Data sets can also consist of a collection of documents or files.

Pre-processing: Pre-processor, a program that processes its input data to produce output that is used as input to another program like a compiler. Data preprocessing, used in machine learning and data mining to make input data easier to work.

Training: The process of training an ML model involves providing an ML algorithm (that is, the learning algorithm) with training data to learn from. The term ML model refers to the model artifact that is created by the training process. You can use the ml model to get predictions on new data for which you do not know the target.

Testing/ Prediction: Machine Learning (models) represents a class of software that learns from a given set of data and then makes predictions on the new data set based on its learning. In other words, the Machine Learning models are trained with an existing data set in order to make the prediction on a new data set.

Prediction: is the output of an algorithm after it has been trained on a historical dataset and applied to new data when you're trying to forecast the likelihood of a particular outcome, such as whether or not a customer will churn in 30 days.

Evaluation of Models / Comparison: Model Evaluation is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future. ... To avoid over fitting, both methods use a test set (not seen by the model) to evaluate model performance Approaches to comparing models. Which model is 'best'? But that approach is too simple when the models have different numbers of parameters, which is usually the case. Prism offers two approaches to comparing models with different numbers of parameters.

**3.2 FEATURES OF PROPOSED SYSTEM**

* Efficient, reliable and flexible counter measures to detect fake news.
* Provides high accuracy to determine a news is fake or true.
* Preserves word order information.
* Deep learning makes it more useful.

**PYTHON 2.7**

It is the python 2.7 version we are currently implementing in our project. The main reason to select python is because it supports run time typing and also because of all the features it provides and python is one of the stable versions of python.

**3.3 REQUIREMENTS SPECIFICATION**

System analyst tasks to a variety of persons to gather details about the business process and their opinions of why things happen as they do and their ideas for changing the process. These can be done through questionnaires, details investigation, observation, collection of samples etc. As the details are collected, the analyst studies the requirements data to identify the features

the new system should have, including both the information the system produces and operational features such as processing controls, response times, and input output methods.

Requirement specification simply means, “Figuring out what to make before you make it”. It determines what people need before you start developing a product for them. Requirement definition is the activity of translating the information gathered in to a document that defines a set of requirements. These should accurately reflect what consumer wants. It is an abstract description of the services that the system should provide and the constraints under the system must operate. This document must be written for that the end user and the stake holder can understand it.

The notations used for requirements definition should be based on natural languages, forms and simple intuitive diagrams. The requirements fall into two categories: functional requirements and non-functional requirements.

The requirements of specification of the proposed system are as follows:

* Python
* Machine Learning

**3.4 FEASIBILITY ANALYSIS**

The feasibility study concern with the considerations made to verify whether the system fit to be developed in all terms. Main objective of feasibility study is to test the technical, social and economic feasibility of developing a system. This is done before developing a system. This is done by investigating the existing system in the area under investigation and generating ideas about the new system.

The feasibility study to be conducted for this project involves:

* Technical Feasibility
* Operational Feasibility
* Economic Feasibility
* Behavioral Feasibility

**3.4.1 Technical Feasibility**

The system must be evaluated from the technical view point first. The assessment of this feasibility must be based on an outline design of the system requirement in terms of input, output, programs, procedure and staff. Having identified the outline of the system, the investigation must go on to suggest the type of equipment, required method of developing the system, and the method of running the system.

The existing system uses Natural Language Processing. ‘Fake News Detection’ system is developed by using front end as Python and back end as Datasets. It is technically feasible and has lots of features. It is currently implemented in the windows 10 platform. But it is also feasible to work in the Linux platform. We use the ANACONDA Navigator (GUI) which is a package and environment manager, which use channels without using command line commands. It also includes Exchangeable Image File Format (EXIF), Conventional Neural Network (CNN), Deep CNN, Machine Learning and Naïve Bayes model. The main reason to prefer Python language is that because it has run time typing and it also is simple and needs only short period of time to execute operations. Machine Learning is basically used for automation and saving time.

**3.4.2 Operational Feasibility**

It is mainly related to human organizational and political aspects. This test of feasibility asks if the system will work when it is developed and installed. It also measures how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development.

There is no difficulty in implementing the system. The ‘Fake News Detection’ is effective and user friendly. To ensure success, desired operational outcomes must be imparted during design and development. These include such design-dependent parameters such as reliability, maintainability, supportability, usability and others.

**3.4.3 Economic Feasibility**

In the economic feasibility the development cost of the system is evaluated weighting it against the ultimate benefit derived from the new system. It is found that the benefit, from the new system would be more than the cost and time involved in its development. This project ‘Fake News Detection’ is economically feasible because IDE used for developing the software is free of cost.

The proposed system uses Python as the front end which is a free and open source software therefore it can be downloaded easily from the internet. Also, the Anaconda GUI and all the other datasets are downloaded from the internet with free of cost.

**3.4.4 Behavioral Feasibility**

The behavioral feasibility depends upon whether the system performed in the expected way or not. Behavioral Feasibility study is a test of system proposal according to it workability, impact on organization, ability to meet user’s need and effective use of resources. However, a feasibility study provides a useful starting point for full analysis. ‘Fake News Detection’ is behaviorally feasible because of the effective use of the resources and also the system satisfies user needs and is user friendly.

The different datasets used will increase the systems performance and makes is more effective. The user will be able to identify the morphed or malicious. There are five modules included in this project they are: Datasets, pre-processing, training, testing and evaluation. With the influence of these modules it makes the project more effective.

**CHAPTER 4**

**OPERATING ENVIRONMENT**

**4.1 HARDWARE REQUIREMENTS**

Processor : Dual Core or above

RAM : 4GB

Hard Disk : 500GB

* 1. **SOFTWARE REQUIREMENTS**

Operating System : Windows 10

Language : Python, Machine Learning

DB : CSV Dataset

Server : Xampp Built-in Server

IDE : Sublime

**4.3 TOOLS AND PLATFORM**

**Python**

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991. When he began implementing Python, Guido van Rossum was also reading the published scripts from [“Monty Python’s Flying Circus”](https://en.wikipedia.org/wiki/Monty_Python), a BBC comedy series from the 1970s. Van Rossum thought he needed a name that was short, unique, and slightly mysterious, so he decided to call the language Python. Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python can be easy to pick up whether you're a first-time programmer or you're experienced with other languages. The [Python Package Index (PyPI)](https://pypi.python.org/) hosts thousands of third-party modules for Python. Both Python's standard library and the community-contributed modules allow for endless possibilities. Python is developed under an OSI-approved open source license, making it freely usable and distributable, even for commercial use. Python's license is administered by the python software foundation. Python is also an interpreted, interactive, language. It incorporates modules, exceptions, dynamic typing, very high-level dynamic data types, and classes. Python combines remarkable power with very clear syntax. It has interfaces to many system calls and libraries, as well as to various window systems, and is extensible in C or C++. It is also usable as an extension language for applications that need a programmable interface. Finally, Python is portable: it runs on many Unix variants, on the Mac, and on Windows 2000 and later.

The Python Software Foundation is an independent non-profit organization that holds the copyright on Python versions 2.1 and newer. The PSF’s mission is to advance open source technology related to the Python programming language and to publicize the use of Python

Python is can be applied to many different classes of problems. The language comes with a large standard library that covers areas such as string processing (regular expressions, Unicode, calculating differences between files), Internet protocols (HTTP, FTP, SMTP, XML-RPC, POP, IMAP, CGI programming), software engineering (unit testing, logging, profiling, parsing Python code), and operating system interfaces (system calls, filesystems, TCP/IP sockets).

Python versions are numbered A.B.C or A.B. A is the major version number – it is only incremented for really major changes in the language. B is the minor version number, incremented for less earth-shattering changes. C is the micro-level – it is incremented for each bugfix release. Not all releases are bugfix releases. In the run-up to a new major release, a series of development releases are made, denoted as alpha, beta, or release candidate. Alphas are early releases in which interfaces aren’t yet finalized; it’s not unexpected to see an interface change between two alpha releases. Betas are more stable, preserving existing interfaces but possibly adding new modules, and release candidates are frozen, making no changes except as needed to fix critical bugs.

Python is very stable. New, stable releases have been coming out roughly every 6 to 18 months since 1991, and this seems likely to continue. Currently there are usually around 18 months between major releases. Python 3.7.4 is the latest version of Python released on 08 July 2019

**FEARURES OF PYTHON**

**Simple Language:**

Coding in Python is very simple. A user with the non-coding background can easily learn this language if they have the good logical skill as python use basic English language for coding.

2. **Free and Open Source:**

Python is a freeware and open source software so it can be easily downloaded and used. It doesn’t require any activation key or subscription to work on it.

3. **Portability:**

Python can run any operating system. Also, we use the python code written on one system onto another system without making any changes to the code.

4. **Extensible and Embeddable:**

This feature is provided to the user as to extend python from its initial state.

An extensible software program, for example, might support add-ons or plug-ins that add extra functionality to the program.

5. **High-level Interpreted Language:**

In this feature the code which execute instruction directly and freely, without previously compiling a program into machine language instructions.

6. **Object Oriented :**

Python is object-oriented so it is organized around objects rather than "actions" and data rather than logic.

***What python can do:***

1. It can be used on a server to create web applications.

2. It can be used to connect to the database system and also perform a various task like read and modify.

3. It can be used to handle big data and perform complex mathematics.

4. It can also be used for data mining process.

5. It can be used for rapid prototyping, or for production-ready software development.

**4.4 Machine Learning**

**Machine Learning** is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that which makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

**Arthur Samuel**, a pioneer in the field of artificial intelligence and computer gaming, coined the term **“Machine Learning”**. He defined machine learning as – **“Field of study that gives computers the capability to learn without being explicitly programmed”**.

*TensorFlow-. It is new neural Network API. It is this ML library which we include in this project.*

In a very layman manner, Machine Learning(ML) can be explained as automating and improving the learning process of computers based on their experiences without being actually programmed i.e. without any human assistance. The process starts with feeding good quality data and then training our machines(computers) by building machine learning models using the data and different algorithms. The choice of algorithms depends on what type of data do we have and what kind of task we are trying to automate. **Example: Training of students during exam.**

**Basic Difference in ML and Traditional Programming?**

**Traditional Programming:** We feed in DATA (Input) + PROGRAM (logic), run it on machine and get output.

**Machine Learning:** We feed in DATA(Input) + Output, run it on machine during training and the machine creates its own program(logic), which can be evaluated while testing.  
**How ML works?**

* Gathering: past data I gathered any form suitable for processing. The better the quality of data, the more suitable it will be for modeling
* Data Processing –Sometimes, the data collected is in the raw form and it needs to be pre-processed.
* Divide the input data into training, cross-validation and test sets. The ratio between the respective sets must be 6:2:2
* Testing our conceptualized model with data which was not fed to the model at the time of training and evaluating its performance using metrics such as F1 score, precision and recall.

**CHAPTER 5**

**DESIGN**

**5.1 SYSTEM DESIGN**

System design is a reduction of an entire system by studying the various operations performed and their relationships within the system and the requirements of its success. One aspect of design is defining the boundaries of the system and determining whether or not the candidate system should consider other related system.

System can be defined, as an orderly grouping of interdependent components can be simple or complex. The most creative and challenging phase of the system life cycle is system design. The term design describes a final system and the process by which it is developed. It refers to the technical specifications that will be applied in implementing the candidate system. It also includes the construction of programs and program testing.

The first step in the system design is to determine how the output is to be produced and in what format. Samples of the output and the inputs are also presented. In the second step, input data and master files are to be designed to meet requirement of the proposed output .The processing phase’s system’s objectives and complete documentation.

System design has two phases:

* Logical
* Physical

The logical design reviews the present physical system, prepares the input and output and also prepares a logical design walk- through. We have to deal with how to take entries required and whether and how to process the user data. Also we have to deal with how to present the data in an informative and appealing format. This design also involves the methodology to store, modify and retrieve data from the data base as per the requirement.

Physical design maps out the details of the physical system, plans the system implementation, devices a test and implementation plan and new hardware and software. We have to decide how and where to store the input data and how to process it so as to present it to the user in an easy, informative and attractive manner.

**5.2 Data Flow Diagram**

A Data Flow Diagram, (DFD) or Bubble chart is s a network that describes the flow of a data and processes that change, or transform data throughout the system. This network is constructed by using a set of symbols that can do not imply a physical implementation. It is a graphical tool for structures analysis of the system requirements. DFD models a system by using external entities from which data flows to a process which transforms data and creates, outputs data flow which goes which goes to other processes or external entities of files. Data in files may also flow to processes as inputs.

DFD’s can be hierarchically organized which help in partitioning and analyzing large systems. As a first step, one DFD can depict an entire system which gives the system overview. It is called Context Diagram or level 0 DFD. The Context diagram can be further expanded. The successive expansion of the DFD from the DFD from the context diagram to those giving more details is known as levelling of DFD. Thus a top down approach is used starting with an overview and working out the details.

The main merit of DFD is that it can provide an overview of the system requirements, what data a system would process, what transformations of data are done, what files are used and where results flow.

**Basic data flow symbols**

A process represents transformation where incoming data flows are changed into outgoing data flows.

A data flow is route, which enables packets of data to travel from one point to another. Data may flow from a source to a processor and from data store or process. An arrow line depicts the flow, with arrow head pointing in the direction of the flow.

A data source is a repository of data that is to be stored for the use by one or more process may be simple as buffer or queue or sophisticated as uses the content of store and does not alter it, the arrow head goes only from the store to the process. If a process alters the details in the store then a double headed arrow is used.

The notations used in this project DFD is given below:

: Represents source or destination of data

: Represents a process

: Represents data flow

: Represent file

: Represent data warehouse

**Project DFD**

**Level 0**

responds

request

User

User

**Userz**

responds

responds

request

responds

request

request

responds

responds

request

request

responds

CNN

Deep-CNN

LSTM

**Level 1 of User**

request

User

User

Test

data

Trained

data

Identify fake news

Testing

Training

Model

Learn

Data

Warehouse

Pre-processor

**5.3 INPUT OUTPUT DESIGN**

The input design is the process of converting the user-oriented description of inputs into a programmer-oriented specification. The objective of input design is to create an input layout that is easy to follow and prevents the user from receiving fake news.

The goal designing input data is to make the detection of fake news easy and free from errors as possible. For providing a good input design for the application easy data input and selection feature and adopted.

Computer output is the most important one to the user. A major form of the output is the display of the information gathered by the system and the servicing the user requests to the system. Output generally refers to the results or information that is generated by the system.

It can be in the form of textual and image. Since some of the users of the system may not operate the system, but merely use the output from the system to aid them in decision-making, much importance is given to the output design.

Currently we are giving textual data and cut off images as input and desired output we are getting is accuracy of textual data and result of real or fake image.

**CHAPTER 6**

**FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS**

**6.1 FUNCTIONAL REQUIREMENTS**

In software engineering, a functional requirement defines a function of a software system or its component. A function is described as a set of inputs, the behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Generally, functional requirements are expressed in the form "system must do requirement ".

Functional requirements for each of the cases described below:

1. The system shall have options for the user to detect both fake news and fake images
2. The system shall provide confidentiality to the user
3. The system provides accuracy details of the news and images.
4. The system provides the user with a vast amount of details of news and images with the help of machine learning algorithms.

**6.2 NON-FUNCTIONAL REQUIREMENTS**

A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviours. Non-functional requirements are “system shall be requirement ". Non-functional requirements are often called qualities of a system. Other terms for non-functional requirements are "constraints", "quality attributes”, “quality goals", "quality of service requirements" and "non-behavioural requirements.

Some of the non-functional requirements are mentioned below:

1. **Usability**: The system shall have a clean interface with only needed features, clear terminology and tool tips wherever necessary. Warnings or alerts shall be specified in clear way.
2. **Efficiency**: The system shall respond to different news being tested.
3. **Portability**: The system shall be independent to some of the specific technological platform used to implement it.
4. **Reliability**: Reliability defined as a measure of the time between failures occurring in a system (measure show frequently the system fails), so that the system shall operate without any failure for a particular period of time
5. **Availability**: Availability measures the percentage of time the system is in its operational state so that the system shall be available for use 24 hours per day and 365days per year. **CHAPTER 7**

**TESTING**

Software testing is critical element of software quality assurance and represents the ultimate review of specifications, design and code generation. System testing is the stage of implementation, it is aimed for ensuring that the system works accurately and efficiently before live operations commences.

Testing is a purpose of executing a programmed with intend of finding errors.

1. Preparing a test case that has high probability of finding undiscovered errors.
2. Testing to erase out all kinds of bucks from the program.

Before going for testing, first we have to decide the type of test. For this impact system, unit testing is carried out. And the following things are taken to consideration.

1. To ensure that information properly places in and out of the program.
2. To ensure that the module operates properly at boundaries established to limit or restrict processing.
3. To find out whether all statements in module have been executed at least once.
4. To find out whether error handling paths are working correctly.

**7.1 TESTING STRATEGIES**

A strategy for software testing integrates software test case design methods in to a well-planned series of steps that results in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project processes. The project manager, software engineer and testing specialists develop a strategy for software testing.

The general characteristics of software testing strategy are:

1. Testing begins at the component level and works “outward” toward the integration of the entire computer system.
2. Different testing techniques are appropriate at different point in time.

A strategy for software testing must accommodate low-level testis that are necessary to verify a small source code segment has been correctly implemented as well as high level testing that validate major system function against customer requirements.

**7.2 WHITE BOX TESTING**

White box testing strategy deals with the internal logic and structure of the code. White box testing is also called as glass, structural, open box or clear box testing. The tests written based on the white box testing strategy incorporate coverage of the code written, branches, paths, statements and internal logic of the code etc. In this project there was many code errors occurred in many web forms, all errors were corrected through debugging.

**CHAPTER 8**

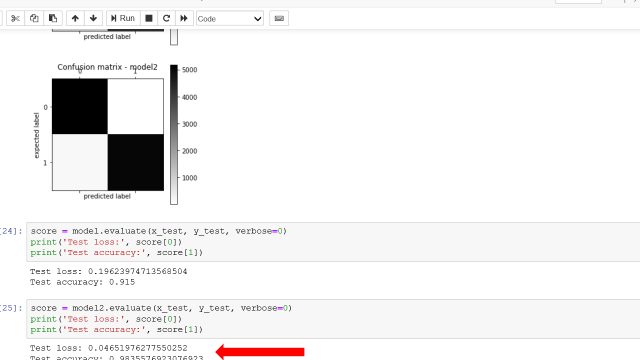
**RESULTS AND DISCUSSION**

We evaluate our models on two closely related tasks: splice detection and splice localization. In the former, our goal is to classify images as being spliced vs. authentic. In the latter, the goal is to localize the spliced regions within an image.

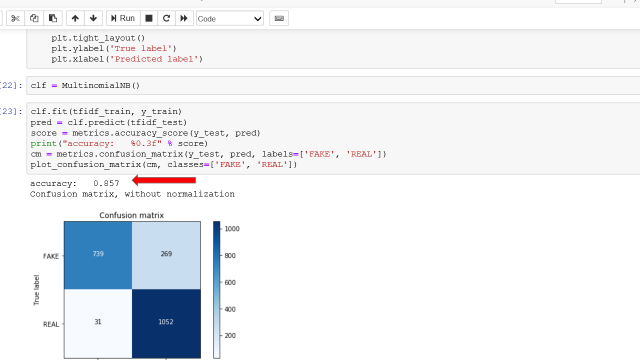
Successfully generated accuracy in textual data and images using Conventional Neural Network and Deep Learning Algorithm.

**8.1 SCREEN SHOTS:**

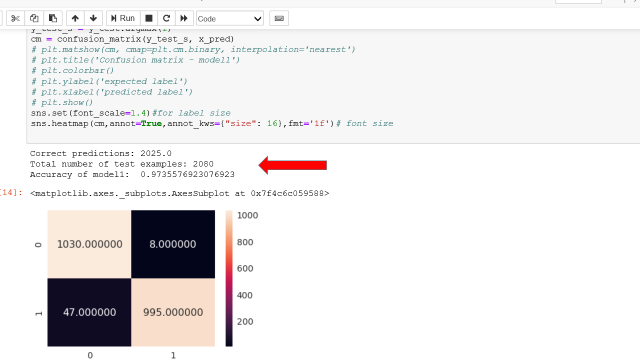
1. CNN Algorithm testing textual data



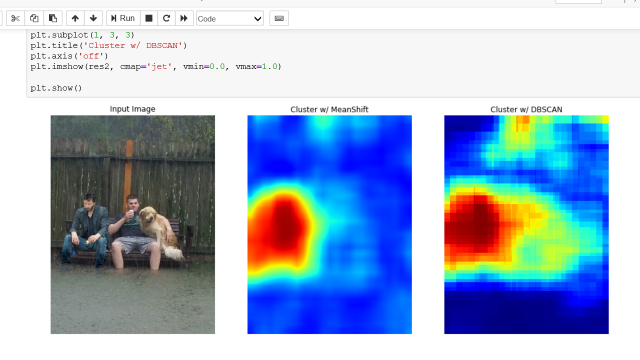
1. Long Short Term Memory Algorithm testing textual data

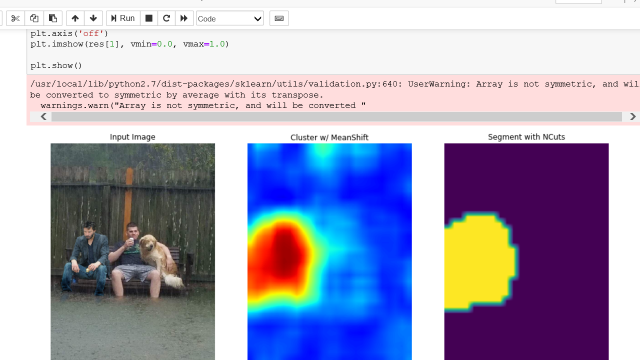


1. ML models Algorithm testing textual data



1. Image segmentation



1. Clustering 

**CHAPTER 9**

**CONCLUSION**

**9.1 SYSTEM IMPLEMENTATION**

Implementation means converting a new design into iteration .During implementation there should be a strong interaction between the developer of the software and the users. Implementation involves installing hardware terminals and training the operating staff. In this phase, user training is critical for minimizing reluctance to change and giving the new system a chance to prove its worth. The new system may be totally new replacing the existing system, or it may be the modifications of existing system. In either case proper implementation is essential to provide a reliable system to meet organizational requirements.

**9.2 CONCLUSION**

With the increasing popularity of social media, more and more people consume news from social media instead of traditional news media. However, social media has also been used to spread fake news, which has strong negative impacts on individual users and broader society. In this article, we explored the fake news problem by reviewing existing literature in two phases: characterization and detection. In the characterization phase, we introduced the basic concepts and principles of fake news in both traditional media and social media. In the detection phase, we reviewed existing fake news detection approaches from a data mining perspective, including feature extraction and model construction. We also further discussed the datasets, evaluation metrics, and promising future directions in fake news detection research and expand the ﬁeld to other applications.

**9.3 FUTURE ENHANCEMENT**

Our goal going forward is to carry out in-depth analyses of our system. The added beneﬁt of our straightforward setup, as opposed to more sophisticated neural network architectures, is that it provides an opportunity to try to understand how it works, what contributes to its performance, and what its limitations are. A particular focus of these analyses will be to try and identify what the mediocre performance of the system with respect to the ‘agree’ and ‘disagree’ labels can potentially be traced back to, next to the limited size of the data set overall and the small number of instances of the labels of speciﬁc interest**.**

**BIBLIOGRAPHY**

Journal paper:

# Evaluating machine learning algorithms for fake news detection

<https://ieeexplore.ieee.org/abstract/document/8305411>

Websites Visited:

<https://docs.python.org/3/faq/general.html#what-is-pythonhttps://docs.python.org/2.7/whatsnew/2.7.html>

<https://www.dezyre.com/article/top-10-machine-learning-projects-for-beginners/397>

<https://www.quora.com/What-are-the-features-of-Python>

<https://docs.python.org/3/faq/general.html#what-is-pythonhttps://docs.python.org/2.7/whatsnew/2.7.html>

<https://www.quora.com/What-are-the-different-versions-of-Python>

<https://www.geeksforgeeks.org/machine-learning/>

https://www.dezyre.com/article/top-10-machine-learning-projects-for-beginners/397

**APPENDICES**

**SCRUM MODEL**

**i. Git**

Git is a version-control system for tracking changes in computer files and coordinating work on those files among multiple people. It is primarily used for source-code management in software development, but it can be used to keep track of changes in any set of files. As a distributed revision-control system, it is aimed at speed, data integrity, and support for distributed, non-linear workflows.

**ii. Git Repositories**

A Git repository contains the history of a collection of files starting from a certain directory. The process of copying an existing Git repository via the Git tooling is called cloning. After cloning a repository the user has the complete repository with its history on his local machine. Of course, Git also supports the creation of new repositories.

If you want to delete a Git repository, you can simply delete the folder which contains the repository. If you clone a Git repository, by default, Git assumes that you want to work in this repository as a user. Git also supports the creation of repositories targeting the usage on a server.

**iii. Scrum**

Scrum is an agile way to manage a project, usually software development. Agile software development with Scrum is often perceived as a methodology; but rather than viewing Scrum as methodology, think of it as a framework for managing a process. In the agile Scrum world, instead of providing complete, detailed descriptions of how everything is to be done on a project, much of it is left up to the Scrum software development team. This is because the team will know best how to solve the problem they are presented. In the agile Scrum world, instead of providing complete, detailed descriptions of how everything is to be done on a project, much of it is left up to the Scrum software development team. This is because the team will know best how to solve the problem they are presented.

Within agile development, Scrum teams are supported by two specific roles. The first is a Scrum Master, who can be thought of as a coach for the team, helping team members use the Scrum process to perform at the highest level. The product owner (PO) is the other role, and in Scrum software development, represents the business, customers or users, and guides the team toward building the right product.

PRODUCT BACKLOG

To create research project which facilitates the fake activity taking place in text data and images based on Python platform using Model View Control Architecture by,

* Creation of work area and roles and project.
* Adding modules and adding test cases.
* Developing module contents and test the module.

DAILY SPRINT

DAY 1

Team has been formed with two members.

DAY 2

Topic Submitted to the Scrum Master.

DAY 3

Topic approved by Scrum Master.

DAY 4

Created Git Hub account and abstract is submitted.

DAY 5

Learned basics of Python.

DAY 6

Learned what are the technologies used in ML.

DAY 7

Studied about how ML and Python are interrelated and how they are implemented together.

DAY 8

Installed python and set the path using environmental variable.

System settings--Advance System settings—Environmental variable—Path

DAY 9

Installed IDE(Anaconda 3) package of ML ,also installed XAMPP and also get to know about CSV Comma Separated Value.

DAY 10

Practiced various python programs.

DAY 11

Planned overall module and its working process. Total 5 modules included in this project. The modules are:  
 1. Dataset Collection

2. Pre-processing

3. Training

4. Testing and Predicting

5. Evaluation / Comparing of Models

DAY 12

Collected different kinds of images and texts for the first module

Dataset Collection and stored it for further use.

DAY 13

Created second module Preprocessing and using dataset collection as input preprocessed all the data using preprocessing techniques.

DAY 14

Created a repository in github account and added all the data till the

last work.

DAY 15

Documentation started.

DAY 16

Created third module Training were trained our machine using input and output data for self-learning of machine itself.

DAY 17

Created the next module Testing and Predicting, where testing is done using Trained input data. Predicts the best input to get the best output.

DAY 18

Description of each test has been completed

DAY 19

Continued Testing process using different kinds of data, such as black box and white box testing

DAY20

Quality assurance successfully performed

DAY 21

Added all the updates to the github repository.

DAY 22

Created final module Evaluation/ Comparing using input data which is tested, trained and preprocessed.

DAY 23

In last module Compared each trained data with new data and Evaluated accuracy, integrity and performance of data.

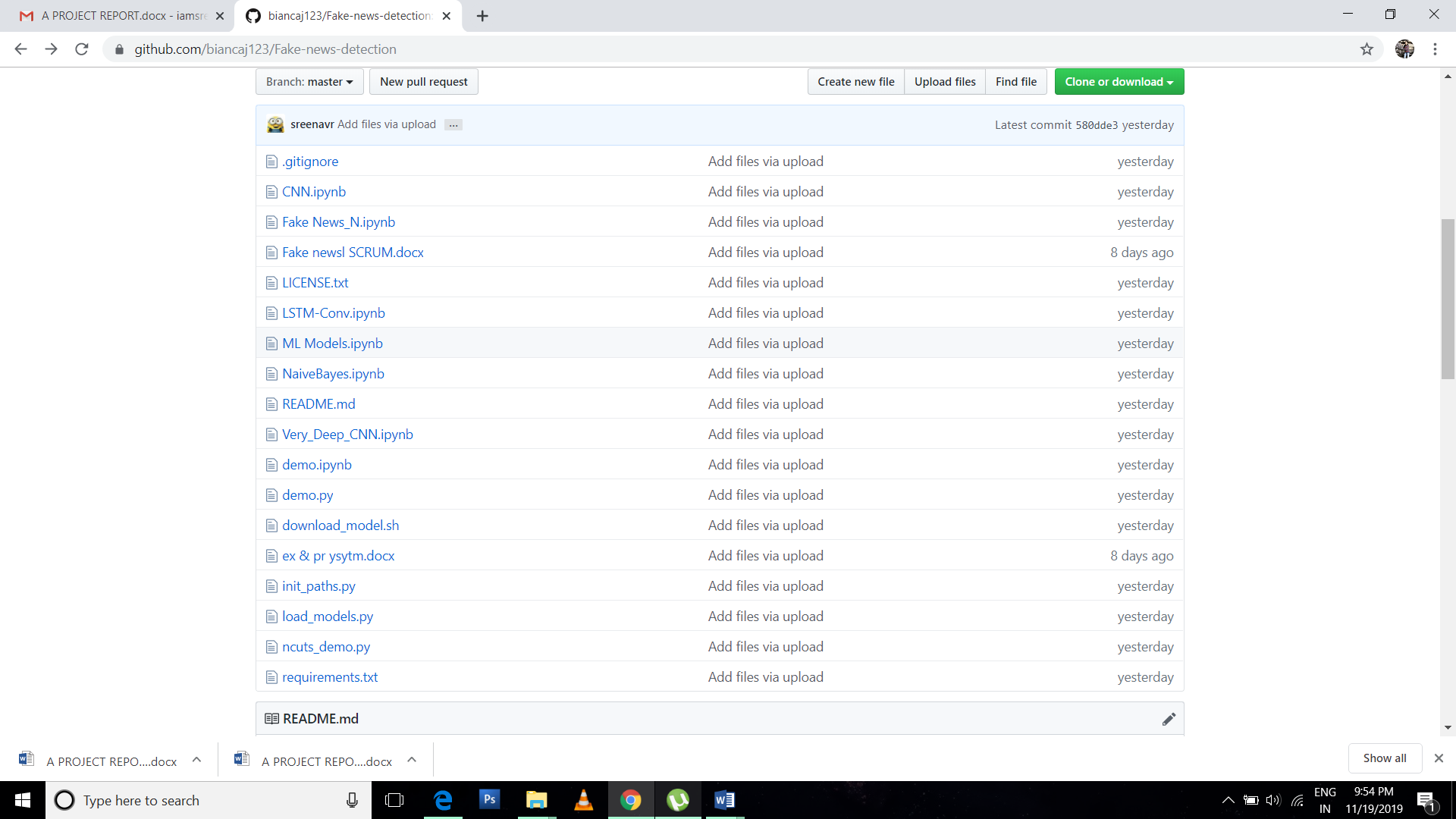
DAY 24

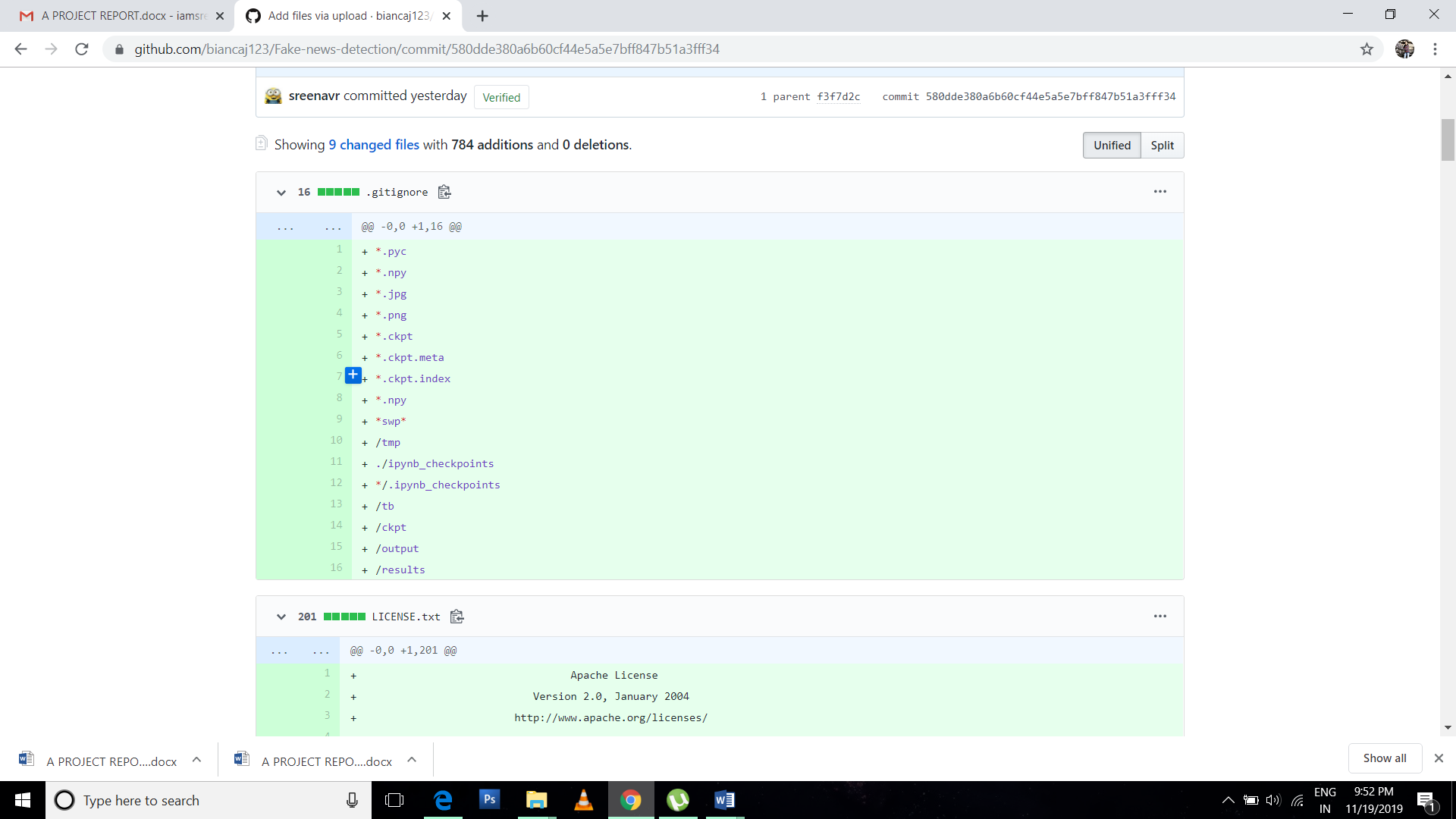
Testing the entire system.

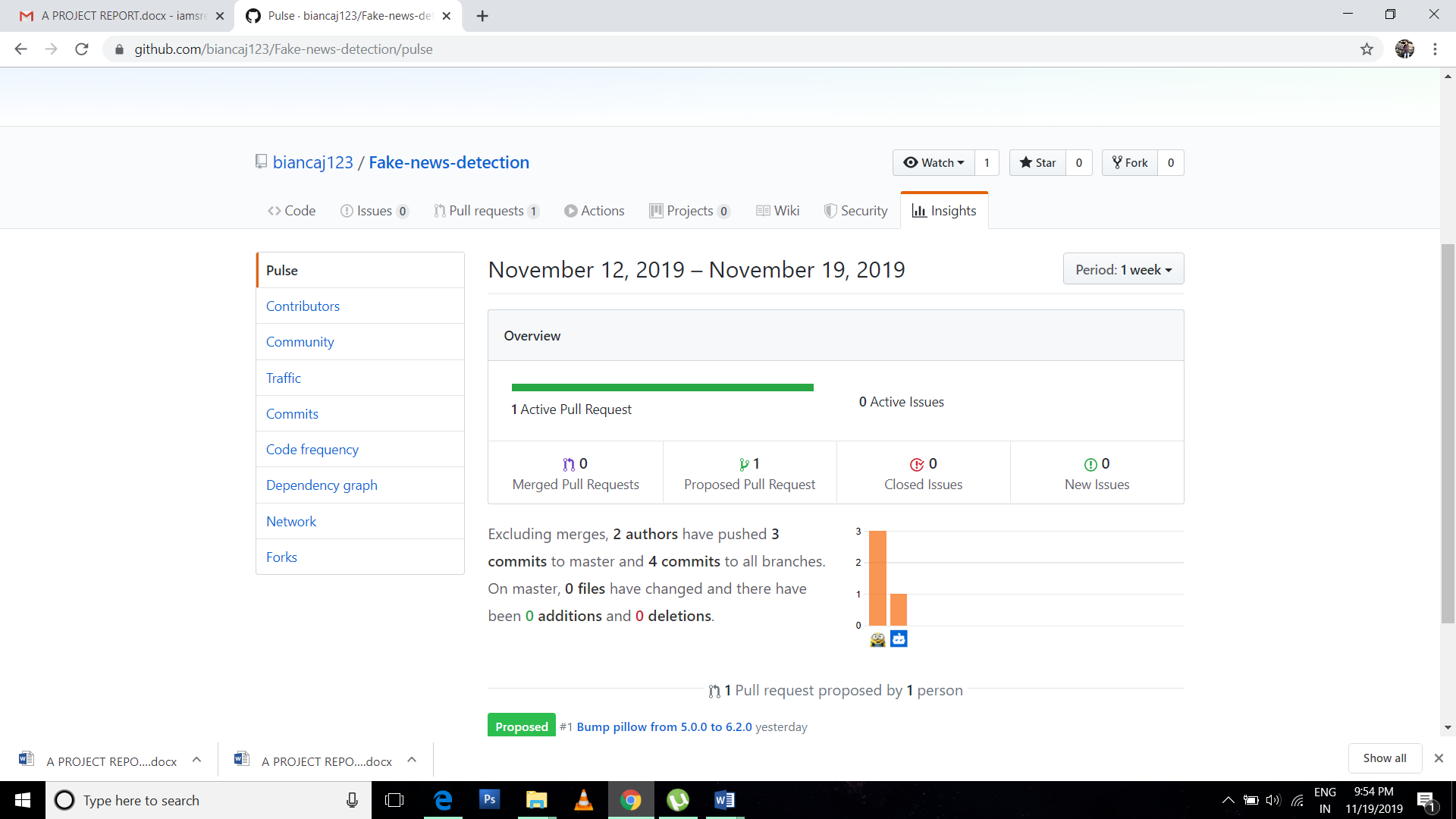
DAY 25

Documentation completed.

**iv.Git History**







**v.Coding**

Very Deep CNN.ipynb

{

"cells": [

{

"cell\_type": "code",

"execution\_count": 1,

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"C:\\ProgramData\\Anaconda3\\lib\\site-packages\\h5py\\\_\_init\_\_.py:36: FutureWarning: Conversion of the second argument of issubdtype from `float` to `np.floating` is deprecated. In future, it will be treated as `np.float64 == np.dtype(float).type`.\n",

" from .\_conv import register\_converters as \_register\_converters\n",

"Using TensorFlow backend.\n"

]

},

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Import Successful\n"

]

}

],

"source": [

"import keras\n",

"from tensorflow.python.client import device\_lib\n",

"\n",

"#print(device\_lib.list\_local\_devices())\n",

"import numpy as np\n",

"import pandas as pd\n",

"import re\n",

"\n",

"\n",

"import os\n",

"\n",

"from keras.preprocessing.text import Tokenizer\n",

"from keras.preprocessing.sequence import pad\_sequences\n",

"from keras.utils.np\_utils import to\_categorical\n",

"\n",

"from keras.layers import Embedding, Activation\n",

"from keras.layers import Dense, Input, Flatten, Bidirectional\n",

"from keras.layers import Conv1D, MaxPooling1D, Embedding, Merge, Dropout\n",

"from keras.models import Model\n",

"\n",

"from keras.models import Sequential\n",

"from keras.layers.convolutional import Conv3D\n",

"from keras.layers.convolutional\_recurrent import ConvLSTM2D\n",

"from keras.layers.normalization import BatchNormalization\n",

"import numpy as np\n",

"from keras.layers import Dense, Embedding, LSTM, GRU\n",

"\n",

"MAX\_SEQUENCE\_LENGTH = 2000\n",

"MAX\_NB\_WORDS = 600000\n",

"EMBEDDING\_DIM = 100\n",

"VALIDATION\_SPLIT = 0.2\n",

"print(\"Import Successful\")"

]

},

{

"cell\_type": "code",

"execution\_count": 2,

"metadata": {},

"outputs": [],

"source": [

"def clean\_str(string):\n",

" \"\"\"\n",

" Cleaning of dataset\n",

" \"\"\"\n",

" string = re.sub(r\"\\\\\", \"\", string) \n",

" string = re.sub(r\"\\'\", \"\", string) \n",

" string = re.sub(r\"\\\"\", \"\", string) \n",

" return string.strip().lower()\n"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": [

"\n"

]

},

{

"cell\_type": "code",

"execution\_count": 3,

"metadata": {},

"outputs": [],

"source": [

"# Input Data preprocessing\n",

"data\_train = pd.read\_csv('data/train\_Mixed.csv')\n",

"#print(data\_train.columns)\n",

"#print('What the raw input data looks like:')\n",

"#print(data\_train[0:5])\n",

"texts = []\n",

"labels = []\n",

"\n",

"for i in range(data\_train.text.shape[0]):\n",

" text1 = data\_train.title[i]\n",

" text2 = data\_train.text[i]\n",

" text = str(text1) +\"\"+ str(text2)\n",

" texts.append(text)\n",

" labels.append(data\_train.label[i])\n",

" \n",

"tokenizer = Tokenizer(num\_words=MAX\_NB\_WORDS)\n",

"tokenizer.fit\_on\_texts(texts)\n",

"sequences = tokenizer.texts\_to\_sequences(texts)\n",

"\n",

"word\_index = tokenizer.word\_index\n",

"print(\"Data Improted\")\n",

"#print('Found %s unique tokens.' % len(word\_index))"

]

},

{

"cell\_type": "code",

"execution\_count": 4,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"preprocess complete\n"

]

}

],

"source": [

"# Pad input sequences\n",

"data = pad\_sequences(sequences, maxlen=MAX\_SEQUENCE\_LENGTH)\n",

"labels = to\_categorical(np.asarray(labels),num\_classes = 2)\n",

"# print('Shape of data tensor:', data.shape)\n",

"# print('Shape of label tensor:', labels.shape)\n",

"print(\"preprocess complete\")"

]

},

{

"cell\_type": "code",

"execution\_count": 5,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"train\_test\_split\_complete\n"

]

}

],

"source": [

"# Train test validation Split\n",

"from sklearn.model\_selection import train\_test\_split\n",

"\n",

"indices = np.arange(data.shape[0])\n",

"np.random.shuffle(indices)\n",

"data = data[indices]\n",

"labels = labels[indices]\n",

"x\_train, x\_test, y\_train, y\_test = train\_test\_split( data, labels, test\_size=0.20, random\_state=42)\n",

"x\_test, x\_val, y\_test, y\_val = train\_test\_split( x\_test, y\_test, test\_size=0.50, random\_state=42)\n",

"# print('Size of train, validation, test:', len(y\_train), len(y\_val), len(y\_test))\n",

"\n",

"# print('real & fake news in train,valt,test:')\n",

"# print(y\_train.sum(axis=0))\n",

"# print(y\_val.sum(axis=0))\n",

"# print(y\_test.sum(axis=0))\n",

"print(\"train\_test\_split\_complete\")"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": [

"\n"

]

},

{

"cell\_type": "code",

"execution\_count": 6,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Glove Complete\n"

]

}

],

"source": [

"#Using Pre-trained word embeddings\n",

"GLOVE\_DIR = \"data\" \n",

"embeddings\_index = {}\n",

"f = open(os.path.join(GLOVE\_DIR, 'glove.6B.100d.txt'), encoding=\"utf8\")\n",

"for line in f:\n",

" values = line.split()\n",

" #print(values[1:])\n",

" word = values[0]\n",

" coefs = np.asarray(values[1:], dtype='float32')\n",

" embeddings\_index[word] = coefs\n",

"f.close()\n",

"\n",

"#print('Total %s word vectors in Glove.' % len(embeddings\_index))\n",

"\n",

"embedding\_matrix = np.random.random((len(word\_index) + 1, EMBEDDING\_DIM))\n",

"for word, i in word\_index.items():\n",

" embedding\_vector = embeddings\_index.get(word)\n",

" if embedding\_vector is not None:\n",

" # words not found in embedding index will be all-zeros.\n",

" embedding\_matrix[i] = embedding\_vector\n",

" \n",

"embedding\_layer = Embedding(len(word\_index) + 1,\n",

" EMBEDDING\_DIM,\n",

" weights=[embedding\_matrix],\n",

" input\_length=MAX\_SEQUENCE\_LENGTH)\n",

"\n",

"print(\"Glove Complete\")"

]

},

{

"cell\_type": "code",

"execution\_count": 7,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Model Compiled\n"

]

}

],

"source": [

"modell = Sequential()\n",

"modell.add(embedding\_layer)\n",

"for i in range(0,2):\n",

" modell.add(Conv1D(filters=1024, kernel\_size=1, padding='same', activation='relu'))\n",

" modell.add(BatchNormalization())\n",

"\n",

"for i in range(0,5):\n",

" modell.add(Conv1D(filters=32, kernel\_size=5, padding='same', activation='relu'))\n",

" modell.add(BatchNormalization())\n",

" modell.add(Activation('relu'))\n",

" \n",

"for i in range(0,5):\n",

" modell.add(Conv1D(filters=64, kernel\_size=3, padding='same', activation='relu'))\n",

" modell.add(BatchNormalization())\n",

" modell.add(Activation('relu'))\n",

" \n",

"for i in range(0,3):\n",

" modell.add(Conv1D(filters=64, kernel\_size=5, padding='same', activation='relu'))\n",

" modell.add(BatchNormalization())\n",

" modell.add(Activation('relu'))\n",

" modell.add(Conv1D(filters=128, kernel\_size=3, padding='same', activation='relu'))\n",

" modell.add(BatchNormalization())\n",

" modell.add(MaxPooling1D(pool\_size=2))\n",

" modell.add(Activation('relu'))\n",

" \n",

"for i in range (0,7):\n",

" modell.add(Conv1D(filters=128, kernel\_size=5, padding='same', activation='relu'))\n",

" modell.add(BatchNormalization())\n",

" modell.add(Activation('relu'))\n",

"for i in range (0,5):\n",

" modell.add(Conv1D(filters=256, kernel\_size=3, padding='same', activation='relu'))\n",

" modell.add(BatchNormalization())\n",

" modell.add(Activation('relu'))\n",

" \n",

"for i in range (0,3):\n",

" modell.add(Conv1D(filters=256, kernel\_size=5, padding='same', activation='relu'))\n",

" modell.add(BatchNormalization())\n",

" \n",

"for i in range (0,5):\n",

" modell.add(Conv1D(filters=512, kernel\_size=3, padding='same', activation='relu'))\n",

" modell.add(BatchNormalization())\n",

" modell.add(Dropout(0.1))\n",

" \n",

"for i in range(0,2):\n",

" modell.add(Conv1D(filters=768, kernel\_size=5, padding='same', activation='relu')) \n",

" modell.add(BatchNormalization())\n",

" modell.add(MaxPooling1D(pool\_size=2))\n",

" modell.add(Activation('relu'))\n",

"for i in range(0,2):\n",

" modell.add(Conv1D(filters=1024, kernel\_size=3, padding='same', activation='relu'))\n",

" modell.add(BatchNormalization())\n",

" modell.add(Activation('relu'))\n",

" \n",

"#modell.add(Bidirectional(LSTM(512, dropout=0.2, recurrent\_dropout=0.2)))\n",

"modell.add(Dense(1024, activation='relu'))\n",

"modell.add(Dense(512, activation='relu'))\n",

"modell.add(Dense(128, activation='relu'))\n",

"modell.add(Dense(2, activation='softmax'))\n",

"modell.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['accuracy'])\n",

"print(\"Model Compiled\")\n",

"\n"

]

},

{

"cell\_type": "code",

"execution\_count": 8,

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Fitting\n",

"Epoch 1/1\n",

" 8/16640 [..............................] - ETA: 18:04:39 - loss: 0.6570 - acc: 0.7500"

]

},

{

"ename": "KeyboardInterrupt",

"evalue": "",

"output\_type": "error",

"traceback": [

"\u001b[1;31m---------------------------------------------------------------------------\u001b[0m",

"\u001b[1;31mKeyboardInterrupt\u001b[0m Traceback (most recent call last)",

"\u001b[1;32m<ipython-input-8-3743fb647eb6>\u001b[0m in \u001b[0;36m<module>\u001b[1;34m()\u001b[0m\n\u001b[0;32m 1\u001b[0m \u001b[0mprint\u001b[0m\u001b[1;33m(\u001b[0m\u001b[1;34m\"Fitting\"\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[1;32m----> 2\u001b[1;33m \u001b[0mmodell\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0mfit\u001b[0m\u001b[1;33m(\u001b[0m\u001b[0mx\_train\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0my\_train\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0mepochs\u001b[0m\u001b[1;33m=\u001b[0m\u001b[1;36m1\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0mbatch\_size\u001b[0m\u001b[1;33m=\u001b[0m\u001b[1;36m8\u001b[0m\u001b[1;33m,\u001b[0m\u001b[0mverbose\u001b[0m\u001b[1;33m=\u001b[0m\u001b[1;36m1\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0m\u001b[0;32m 3\u001b[0m \u001b[0mprint\u001b[0m\u001b[1;33m(\u001b[0m\u001b[1;34m\"Saving\"\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 4\u001b[0m \u001b[0mmodell\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0msave\u001b[0m\u001b[1;33m(\u001b[0m\u001b[1;34m'VDCNN.h5'\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 5\u001b[0m \u001b[0mprint\u001b[0m\u001b[1;33m(\u001b[0m\u001b[1;34m\"Model Saved\"\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n",

"\u001b[1;32mC:\\ProgramData\\Anaconda3\\lib\\site-packages\\keras\\models.py\u001b[0m in \u001b[0;36mfit\u001b[1;34m(self, x, y, batch\_size, epochs, verbose, callbacks, validation\_split, validation\_data, shuffle, class\_weight, sample\_weight, initial\_epoch, steps\_per\_epoch, validation\_steps, \*\*kwargs)\u001b[0m\n\u001b[0;32m 1000\u001b[0m \u001b[0minitial\_epoch\u001b[0m\u001b[1;33m=\u001b[0m\u001b[0minitial\_epoch\u001b[0m\u001b[1;33m,\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1001\u001b[0m \u001b[0msteps\_per\_epoch\u001b[0m\u001b[1;33m=\u001b[0m\u001b[0msteps\_per\_epoch\u001b[0m\u001b[1;33m,\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[1;32m-> 1002\u001b[1;33m validation\_steps=validation\_steps)\n\u001b[0m\u001b[0;32m 1003\u001b[0m \u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1004\u001b[0m def evaluate(self, x=None, y=None,\n",

"\u001b[1;32mC:\\ProgramData\\Anaconda3\\lib\\site-packages\\keras\\engine\\training.py\u001b[0m in \u001b[0;36mfit\u001b[1;34m(self, x, y, batch\_size, epochs, verbose, callbacks, validation\_split, validation\_data, shuffle, class\_weight, sample\_weight, initial\_epoch, steps\_per\_epoch, validation\_steps, \*\*kwargs)\u001b[0m\n\u001b[0;32m 1703\u001b[0m \u001b[0minitial\_epoch\u001b[0m\u001b[1;33m=\u001b[0m\u001b[0minitial\_epoch\u001b[0m\u001b[1;33m,\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1704\u001b[0m \u001b[0msteps\_per\_epoch\u001b[0m\u001b[1;33m=\u001b[0m\u001b[0msteps\_per\_epoch\u001b[0m\u001b[1;33m,\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[1;32m-> 1705\u001b[1;33m validation\_steps=validation\_steps)\n\u001b[0m\u001b[0;32m 1706\u001b[0m \u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1707\u001b[0m def evaluate(self, x=None, y=None,\n",

"\u001b[1;32mC:\\ProgramData\\Anaconda3\\lib\\site-packages\\keras\\engine\\training.py\u001b[0m in \u001b[0;36m\_fit\_loop\u001b[1;34m(self, f, ins, out\_labels, batch\_size, epochs, verbose, callbacks, val\_f, val\_ins, shuffle, callback\_metrics, initial\_epoch, steps\_per\_epoch, validation\_steps)\u001b[0m\n\u001b[0;32m 1234\u001b[0m \u001b[0mins\_batch\u001b[0m\u001b[1;33m[\u001b[0m\u001b[0mi\u001b[0m\u001b[1;33m]\u001b[0m \u001b[1;33m=\u001b[0m \u001b[0mins\_batch\u001b[0m\u001b[1;33m[\u001b[0m\u001b[0mi\u001b[0m\u001b[1;33m]\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0mtoarray\u001b[0m\u001b[1;33m(\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1235\u001b[0m \u001b[1;33m\u001b[0m\u001b[0m\n\u001b[1;32m-> 1236\u001b[1;33m \u001b[0mouts\u001b[0m \u001b[1;33m=\u001b[0m \u001b[0mf\u001b[0m\u001b[1;33m(\u001b[0m\u001b[0mins\_batch\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0m\u001b[0;32m 1237\u001b[0m \u001b[1;32mif\u001b[0m \u001b[1;32mnot\u001b[0m \u001b[0misinstance\u001b[0m\u001b[1;33m(\u001b[0m\u001b[0mouts\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0mlist\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1238\u001b[0m \u001b[0mouts\u001b[0m \u001b[1;33m=\u001b[0m \u001b[1;33m[\u001b[0m\u001b[0mouts\u001b[0m\u001b[1;33m]\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n",

"\u001b[1;32mC:\\ProgramData\\Anaconda3\\lib\\site-packages\\keras\\backend\\tensorflow\_backend.py\u001b[0m in \u001b[0;36m\_\_call\_\_\u001b[1;34m(self, inputs)\u001b[0m\n\u001b[0;32m 2480\u001b[0m \u001b[0msession\u001b[0m \u001b[1;33m=\u001b[0m \u001b[0mget\_session\u001b[0m\u001b[1;33m(\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 2481\u001b[0m updated = session.run(fetches=fetches, feed\_dict=feed\_dict,\n\u001b[1;32m-> 2482\u001b[1;33m \*\*self.session\_kwargs)\n\u001b[0m\u001b[0;32m 2483\u001b[0m \u001b[1;32mreturn\u001b[0m \u001b[0mupdated\u001b[0m\u001b[1;33m[\u001b[0m\u001b[1;33m:\u001b[0m\u001b[0mlen\u001b[0m\u001b[1;33m(\u001b[0m\u001b[0mself\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0moutputs\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m]\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 2484\u001b[0m \u001b[1;33m\u001b[0m\u001b[0m\n",

"\u001b[1;32mC:\\ProgramData\\Anaconda3\\lib\\site-packages\\tensorflow\\python\\client\\session.py\u001b[0m in \u001b[0;36mrun\u001b[1;34m(self, fetches, feed\_dict, options, run\_metadata)\u001b[0m\n\u001b[0;32m 898\u001b[0m \u001b[1;32mtry\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 899\u001b[0m result = self.\_run(None, fetches, feed\_dict, options\_ptr,\n\u001b[1;32m--> 900\u001b[1;33m run\_metadata\_ptr)\n\u001b[0m\u001b[0;32m 901\u001b[0m \u001b[1;32mif\u001b[0m \u001b[0mrun\_metadata\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 902\u001b[0m \u001b[0mproto\_data\u001b[0m \u001b[1;33m=\u001b[0m \u001b[0mtf\_session\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0mTF\_GetBuffer\u001b[0m\u001b[1;33m(\u001b[0m\u001b[0mrun\_metadata\_ptr\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n",

"\u001b[1;32mC:\\ProgramData\\Anaconda3\\lib\\site-packages\\tensorflow\\python\\client\\session.py\u001b[0m in \u001b[0;36m\_run\u001b[1;34m(self, handle, fetches, feed\_dict, options, run\_metadata)\u001b[0m\n\u001b[0;32m 1133\u001b[0m \u001b[1;32mif\u001b[0m \u001b[0mfinal\_fetches\u001b[0m \u001b[1;32mor\u001b[0m \u001b[0mfinal\_targets\u001b[0m \u001b[1;32mor\u001b[0m \u001b[1;33m(\u001b[0m\u001b[0mhandle\u001b[0m \u001b[1;32mand\u001b[0m \u001b[0mfeed\_dict\_tensor\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1134\u001b[0m results = self.\_do\_run(handle, final\_targets, final\_fetches,\n\u001b[1;32m-> 1135\u001b[1;33m feed\_dict\_tensor, options, run\_metadata)\n\u001b[0m\u001b[0;32m 1136\u001b[0m \u001b[1;32melse\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1137\u001b[0m \u001b[0mresults\u001b[0m \u001b[1;33m=\u001b[0m \u001b[1;33m[\u001b[0m\u001b[1;33m]\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n",

"\u001b[1;32mC:\\ProgramData\\Anaconda3\\lib\\site-packages\\tensorflow\\python\\client\\session.py\u001b[0m in \u001b[0;36m\_do\_run\u001b[1;34m(self, handle, target\_list, fetch\_list, feed\_dict, options, run\_metadata)\u001b[0m\n\u001b[0;32m 1314\u001b[0m \u001b[1;32mif\u001b[0m \u001b[0mhandle\u001b[0m \u001b[1;32mis\u001b[0m \u001b[1;32mNone\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1315\u001b[0m return self.\_do\_call(\_run\_fn, feeds, fetches, targets, options,\n\u001b[1;32m-> 1316\u001b[1;33m run\_metadata)\n\u001b[0m\u001b[0;32m 1317\u001b[0m \u001b[1;32melse\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1318\u001b[0m \u001b[1;32mreturn\u001b[0m \u001b[0mself\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0m\_do\_call\u001b[0m\u001b[1;33m(\u001b[0m\u001b[0m\_prun\_fn\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0mhandle\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0mfeeds\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0mfetches\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n",

"\u001b[1;32mC:\\ProgramData\\Anaconda3\\lib\\site-packages\\tensorflow\\python\\client\\session.py\u001b[0m in \u001b[0;36m\_do\_call\u001b[1;34m(self, fn, \*args)\u001b[0m\n\u001b[0;32m 1320\u001b[0m \u001b[1;32mdef\u001b[0m \u001b[0m\_do\_call\u001b[0m\u001b[1;33m(\u001b[0m\u001b[0mself\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0mfn\u001b[0m\u001b[1;33m,\u001b[0m \u001b[1;33m\*\u001b[0m\u001b[0margs\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1321\u001b[0m \u001b[1;32mtry\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[1;32m-> 1322\u001b[1;33m \u001b[1;32mreturn\u001b[0m \u001b[0mfn\u001b[0m\u001b[1;33m(\u001b[0m\u001b[1;33m\*\u001b[0m\u001b[0margs\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0m\u001b[0;32m 1323\u001b[0m \u001b[1;32mexcept\u001b[0m \u001b[0merrors\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0mOpError\u001b[0m \u001b[1;32mas\u001b[0m \u001b[0me\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1324\u001b[0m \u001b[0mmessage\u001b[0m \u001b[1;33m=\u001b[0m \u001b[0mcompat\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0mas\_text\u001b[0m\u001b[1;33m(\u001b[0m\u001b[0me\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0mmessage\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n",

"\u001b[1;32mC:\\ProgramData\\Anaconda3\\lib\\site-packages\\tensorflow\\python\\client\\session.py\u001b[0m in \u001b[0;36m\_run\_fn\u001b[1;34m(feed\_dict, fetch\_list, target\_list, options, run\_metadata)\u001b[0m\n\u001b[0;32m 1305\u001b[0m \u001b[0mself\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0m\_extend\_graph\u001b[0m\u001b[1;33m(\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1306\u001b[0m return self.\_call\_tf\_sessionrun(\n\u001b[1;32m-> 1307\u001b[1;33m options, feed\_dict, fetch\_list, target\_list, run\_metadata)\n\u001b[0m\u001b[0;32m 1308\u001b[0m \u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1309\u001b[0m \u001b[1;32mdef\u001b[0m \u001b[0m\_prun\_fn\u001b[0m\u001b[1;33m(\u001b[0m\u001b[0mhandle\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0mfeed\_dict\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0mfetch\_list\u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n",

"\u001b[1;32mC:\\ProgramData\\Anaconda3\\lib\\site-packages\\tensorflow\\python\\client\\session.py\u001b[0m in \u001b[0;36m\_call\_tf\_sessionrun\u001b[1;34m(self, options, feed\_dict, fetch\_list, target\_list, run\_metadata)\u001b[0m\n\u001b[0;32m 1407\u001b[0m return tf\_session.TF\_SessionRun\_wrapper(\n\u001b[0;32m 1408\u001b[0m \u001b[0mself\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0m\_session\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0moptions\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0mfeed\_dict\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0mfetch\_list\u001b[0m\u001b[1;33m,\u001b[0m \u001b[0mtarget\_list\u001b[0m\u001b[1;33m,\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[1;32m-> 1409\u001b[1;33m run\_metadata)\n\u001b[0m\u001b[0;32m 1410\u001b[0m \u001b[1;32melse\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0;32m 1411\u001b[0m \u001b[1;32mwith\u001b[0m \u001b[0merrors\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0mraise\_exception\_on\_not\_ok\_status\u001b[0m\u001b[1;33m(\u001b[0m\u001b[1;33m)\u001b[0m \u001b[1;32mas\u001b[0m \u001b[0mstatus\u001b[0m\u001b[1;33m:\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n",

"\u001b[1;31mKeyboardInterrupt\u001b[0m: "

]

}

],

"source": [

"print(\"Fitting\")\n",

"modell.fit(x\_train, y\_train, epochs=3, batch\_size=32,verbose=1)\n",

"print(\"Saving\")\n",

"modell.save('VDCNN.h5')\n",

"print(\"Model Saved\")"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

}

],

"metadata": {

"kernelspec": {

"display\_name": "Python 3",

"language": "python",

"name": "python3"

},

"language\_info": {

"codemirror\_mode": {

"name": "ipython",

"version": 3

},

"file\_extension": ".py",

"mimetype": "text/x-python",

"name": "python",

"nbconvert\_exporter": "python",

"pygments\_lexer": "ipython3",

"version": "3.6.9"

}

},

"nbformat": 4,

"nbformat\_minor": 2

}